ELSEVIER

Discrete Applied Mathematics 98 (2000) 209–217

DISCRETE APPLIED MATHEMATICS

Minimizing the number of late jobs for the two-machine unit-time job-shop scheduling problem ☆

Svetlana A. Kravchenko *

Institute of Engineering Cybernetics, Surganova St. 6, 220012 Minsk, Belarus

Received 11 June 1997; revised 15 February 1999; accepted 15 March 1999

Abstract

This paper considers the problem of minimizing the weighted number of late jobs in the two-machine job-shop with unit time operations. We show its NP-hardness and propose an $O(n^5 \sum w_i)$ time algorithm, which is based on solving optimally a related problem where a maximal set of early jobs is to be determined.

Keywords: Job-shop; NP-hard problem; Pseudo-polynomial algorithm

1. Introduction

The present paper deals with the two-machine unit-time job-shop scheduling problem denoted by $J2 \mid t_{ij} = 1 \mid \sum w_i U_i$. The motivation to study this problem comes mainly from the following results. Lenstra and Rinnooy Kan [10] show that $J2 \mid t_{ij} \in \{1,2\} \mid C_{\max}$ and $J3 \mid t_{ij} = 1 \mid C_{\max}$ are NP-hard in the strong sense. Gonzalez and Sahni [3] do the same for $J2 \mid t_{ij} \in \{0,1\} \mid C_{\max}$ by proving NP-hardness for $J2 \mid \text{pmtn} \mid C_{\max}$. Hefetz and Adiri [4] solve $J2 \mid t_{ij} = 1 \mid C_{\max}$ in time linear in the total number of operations, i.e. in pseudo-polynomial time. Timkovsky [11] and Kubiak et al. [7] give polynomial algorithm for the same problem. Brucker [1,2] extends the result of Hefetz and Adiri [4] to $J2 \mid t_{ij} = 1 \mid L_{\max}$. From the elementary reductions between scheduling problems (see Lawler et al. [9]), it follows immediately that the problems $J2 \mid t_{ij} \in \{0,1\} \mid \sum U_i$, $J2 \mid t_{ij} \in \{1,2\} \mid \sum U_i$ and $J3 \mid t_{ij} = 1 \mid \sum U_i$ are unary NP-hard. Therefore $J2 \mid t_{ij} = 1 \mid \sum U_i$ is the maximal open problem with respect to the minimal number of late jobs criterion. In this paper we give an efficient algorithm for this problem. We show also that $J2 \mid t_{ij} = 1 \mid \sum w_i U_i$ is binary NP-hard and propose a pseudo-polynomial algorithm for this problem.

☆ Supported by the International Association for the Promotion of Cooperation with Scientists from the Independent States of the Former Soviet Union, Project INTAS-93-257.

* Fax: +375-172-318-403.

E-mail address: kravch@newman.bas-net.by (S.A. Kravchenko)

0166-218X/00/$ - see front matter
PII: S0166-218X(99)00165-1

The two-machine unit-time job-shop scheduling model considered in this paper can be formulated as follows. Suppose we are given two machines denoted by A and B, and a set $J = \{J_1, \ldots, J_n\}$ of n jobs to be processed on those machines. Each machine can process at the most one job at a time and each job $J_i \in J$ consists of a chain $(O_{i1}, \ldots, O_{in_i})$ of n_i unit-time operations which have to be processed in this order. Operation O_{ij} has to be processed on machine $\mu_{ij} \in \{A, B\}$, any two consecutive operations O_{ij} and $O_{i,j+1}$ being processed on different machines $\mu_{ij} \neq \mu_{i,j+1}$. Associated with each job J_i there is a due date d_i and a weight w_i. In a given schedule, job J_i finishing by time d_i is called early or on time, otherwise, J_i is called late. We define $U_i = 0$ if J_i is early or on time and $U_i = 1$ if J_i is late. Thus, late job J_i contributes $w_i U_i$ to the weighted number of late jobs. The objective is to schedule the operations so that the total weighted number of late jobs is minimized.

It is easy to see that without changing the $\sum w_i U_i$ value it is possible to replace each noninteger due date d_i by the greatest integer $\lfloor d_i \rfloor$ not exceeding d_i. Thus, all due dates we consider in the paper are integers.

The organization of this paper is as follows. In Section 2 we give some preliminary results. Section 3 presents a pseudo-polynomial algorithm for $J2 \mid t_{ij}=1 \mid \sum w_i U_i$ which turns out to be polynomial for $J2 \mid t_{ij} = 1 \mid \sum U_i$. We also prove NP-hardness for the former problem.

2. Preliminary results and notation

Now we consider the problem of finding the maximal set of early jobs in the two-machine unit-time job-shop scheduling model. The optimal schedule then consists of the set of jobs which meet their due dates and is scheduled first, followed by the set of remaining jobs which do not meet their due dates and is scheduled last. Now we give some alternative form of scheduling model and justify their equivalence.

Let $d = \max\{d_i \mid i \in \{1, \ldots, n\}\}$ be the maximal due date and $z \in \{1, \ldots, d\}$. Associate with each operation O_{ij}, the label $\gamma_{ij} = d_i - n_i + j$ which can be interpreted as the due date of O_{ij}. For any set of jobs $E \subseteq J$ set $E(A,z) = |\{O_{ij} \mid \gamma_{ij} \leqslant z, \mu_{ij} = A, J_i \in E\}|$. So, we define $E(A,z)$ to be the number of operations in E, having the property that each operation has due date not greater than z and it is processed by machine A. In a similar way, we denote by $E(B,z)$ the number of operations in E having a due date not greater than z and being processed by machine B, i.e. $E(B,z) = |\{O_{ij} \mid \gamma_{ij} \leqslant z, \mu_{ij} = B, J_i \in E\}|$.

Now we can prove the following theorem.

Theorem 1. *A set of jobs $E \subseteq J$ can be completed on time if and only if $E(A,z) \leqslant z$ and $E(B,z) \leqslant z$ hold for all $z \in \{1, \ldots, d\}$.*

Proof. *Necessity*: Suppose we have some schedule where all jobs E meet their due dates. Then all operations $\{O_{ij} \mid \gamma_{ij} \leqslant z, \mu_{ij} = A, J_i \in E\}$ are scheduled in the time intervals $[0,1]$, $[1,2], \ldots, [z-1, z]$ and therefore $E(A,z) \leqslant z$ holds for all $z \in \{1, \ldots, d\}$. The proof for $E(B,z)$ is completely analogous to that for $E(A,z)$.

Sufficiency: The sufficiency can be proved by induction on d. It is trivially true for $d=1$. Assuming the result for $d=k-1$, we remove two operations, if any, O_{x1} and O_{y1} with the smallest values of γ_{x1} and γ_{y1}, provided that $J_x \in E$, $J_y \in E$ and $\mu_{x1}=A$, $\mu_{y1}=B$ hold. Set $d_i=d_i-1$ for all jobs $J_i \in E$ and denote the modified set E by D. Without the loss of generality we suppose that $\gamma_{x1} \leqslant \gamma_{y1}$ holds. Then, due to the facts that there are only two machines and there are not machine repetitions, i.e. $\mu_{ij} \neq \mu_{i,j+1}$ for all i and j, the following holds: $D(A,z)=E(A,z+1)-1$ for all $z \in \{1,\ldots,k-1\}$; $D(B,z)=E(B,z+1) \leqslant E(A,z)$ for all $z \in \{1,\ldots,\gamma_{y1}-1\}$; $D(B,z)=E(B,z+1)-1$ for all $z \in \{\gamma_{y1},\ldots,k-1\}$. These imply inequalities $D(A,z) \leqslant z$ and $D(B,z) \leqslant z$ for all $z \in \{1,\ldots,k-1\}$. Because of the induction assumption, set D can be completed on time. Now, shifting the obtained schedule to the right by one time unit and scheduling operations O_{x1} and O_{y1} in the time interval $[0,1]$ we obtain the new schedule where all jobs $J_i \in E$ are early. Therefore sufficiency also holds for $d=k$. □

Thus, the problem of finding the maximal set of early jobs is reduced to the problem of finding the maximal set $E \subseteq J$, provided that $E(A,z) \leqslant z$ and $E(B,z) \leqslant z$ hold for all $z \in \{1,\ldots,d\}$. For convenience reformulate this problem in the vector form, i.e. when each job J_i is represented by the vector $J_i=(J_i(A),J_i(B))$, where $J_i(A)=(J_i(A,1),\ldots,J_i(A,d))$ and $J_i(B)=(J_i(B,1),\ldots,J_i(B,d))$, and $J_i(A,z)=|\{O_{ij} \mid j \in \{1,\ldots,n_i\}, \mu_{ij}=A, \gamma_{ij} \leqslant z\}|$ and $J_i(B,z)=|\{O_{ij} \mid j \in \{1,\ldots,n_i\}, \mu_{ij}=B, \gamma_{ij} \leqslant z\}|$ hold for all $z \in \{1,\ldots,d\}$, $i \in \{1,\ldots,n\}$. It is easy to see that in a unique fashion each job J_i is defined by the vector couple $(J_i(A), J_i(B))$. Further we identify these terms. Now in the vector fashion the considered problem sounds:

Find the maximal weighted set of vectors $J_i=(J_i(A),J_i(B))$ whose sum does not exceed vector $C=(CA,CB)$, where $CA=(1,\ldots,d)$, $CB=(1,\ldots,d)$.

In Section 3 we solve the last problem.

Now we introduce some further notation. Denote by $L_i(A)$ the number of nonzero components in vector $J_i(A)$, by $D_i(A)$ the number of nonzero components equal to the last one, and by $H_i(A)$ the number of nonzero unequal components in vector $J_i(A)$. Formally one can write

$$L_i(A)=|\{z \mid J_i(A,z) \neq 0, z \in \{1,\ldots,d\}\}|,$$

$$D_i(A)=|\{z \mid J_i(A,z)=J_i(A,d), z \in \{1,\ldots,d\}\}|,$$

$$H_i(A)=|\{O_{ij} \mid \mu_{ij}=A, j \in \{1,\ldots,n_i\}\}|,$$

where $i \in \{1,\ldots,n\}$.

The values $L_i(B)$, $D_i(B)$ and $H_i(B)$ are introduced in a similar way

$$L_i(B)=|\{z \mid J_i(B,z) \neq 0, z \in \{1,\ldots,d\}\}|,$$

$$D_i(B)=|\{z \mid J_i(B,z)=J_i(B,d), z \in \{1,\ldots,d\}\}|,$$

$$H_i(B)=|\{O_{ij} \mid \mu_{ij}=B, j \in \{1,\ldots,n_i\}\}|.$$

In case when for some job $J_i(A,z)=0$ holds for all z, we set $L_i(A)=L_i(B)$, $D_i(A)=L_i(B)$, $H_i(A)=0$. If $J_i(B,z)=0$ holds for all z, then we set $L_i(B)=L_i(A)$, $D_i(B)=L_i(A)$ and $H_i(B)=0$.

It is obvious that with the help of $L_i(A)$, $D_i(A)$ and $H_i(A)$ the vector $J_i(A)$ is reconstructed in a unique manner, because, if $H_i(A)\neq 0$ then

$$J_i(A,z)=0 \quad \text{for all } z\in\{1,\dots,d-L_i(A)\},$$

$$J_i(A,z+2)=J_i(A,z)+1 \quad \text{for all } z\in\{d-L_i(A)-1,\dots,d-D_i(A)\},$$

$$J_i(A,z)=H_i(A) \quad \text{for all } z\in\{d-D_i(A)+1,\dots,d\}.$$

In this case $H_i(A)=1+(L_i(A)-D_i(A))/2$ holds, but if $H_i(A)=0$ then $J_i(A,z)=0$ holds for all $z\in\{1,\dots,d\}$. Note that vector $J_i=(J_i(A),J_i(B))$, $i\in\{1,\dots,n\}$, is completely defined by the six values $L_i(A)$, $D_i(A)$, $H_i(A)$, $L_i(B)$, $D_i(B)$, $H_i(B)$.

Before going any further, let us look at an example.

Example 1. The job J_i with $n_i=5, d_i=7, \mu_{i1}=A$ provided that $d=8$ is represented by the couple $J_i(A)=(0,0,1,1,2,2,3,3)$ and $J_i(B)=(0,0,0,1,1,2,2,2)$. Here $L_i(A)=6$, $L_i(B)=5$, $D_i(A)=2$, $D_i(B)=3$, $H_i(A)=3$ and $H_i(B)=2$ hold.

In Section 3 we suppose that all vectors J_i, $i\in\{1,\dots,n\}$, are ordered in nonincreasing order of $\max\{L_i(A),L_i(B)\}$.

3. The algorithm

In general the problem of finding the maximal vector set whose sum is not more than some given vector is extremely hard to be solved optimally. However, in our case all vectors have very special structure. In the next theorem we use this structure to derive a property which permits us to use the dynamic programming approach to solve the problem in question.

Suppose we have some set of vectors $E=\{J_1(A),\dots,J_k(A)\}$ ordered in nonincreasing order of their $\max\{L_i(A),L_i(B)\}$ values. Denote by $FA(E)$ and $PA(E)$ the smallest and the second smallest values of $\{D_i(A),L_i(A)\,|\,J_i\in E\}$. Consider the following

Example 2. Let set E contains three vectors J_1,J_2 and J_3, where $J_1(A)=(0,1,1,2,2,2,2,2)$, $J_2(A)=(0,0,1,1,2,2,3,3)$ and $J_3(A)=(0,0,0,1,1,2,2,3)$ hold. Here $D_1(A)=5$, $D_2(A)=2$, $D_3(A)=1$, $L_1(A)=7$, $L_2(A)=6$, $L_3(A)=5$ hold. Then $FA(E)=D_3(A)=1$ and $PA(E)=D_2(A)=2$.

Let the sum of $J_1(A),\dots,J_{k-1}(A)$ be a feasible vector. We say that a set of jobs is feasible if there exists a schedule for this set in which all jobs are completed on time. In particular, further we suppose that each vector J_i is a feasible one. Taking into account Theorem 1 in this case inequality $\sum_{i=1}^{k-1}J_i(A)\leqslant CA$ holds. Now, to decide whether $\{J_1(A),\dots,J_{k-1}(A)\}\cup\{J_k(A)\}$ can be feasible, one can use the next theorem.

Theorem 2. *The sum of vectors* $E=\{J_1(A),\dots,J_k(A)\}$ *is less or equal to* CA *if and only if the* $(d-PA(E)+1)$*th component of the vector* $\sum_{i=1}^{k} J_i(A)$ *is less or equal to* $CA(d-PA(E)+1)$.

Proof. The necessity is evident.

Sufficiency: The sufficiency is deduced by induction on the number of vectors k. If the number of vectors is 1, then $J_1(A,d-PA(E)+1)=1$ holds due to $PA(E)=L_1(A)$ holds. Therefore $J_1(A,d-PA(E)+1)\leqslant CA(d-PA(E)+1)$ implies $d\geqslant L_1(A)$, which means that $J_1(A)\leqslant CA$.

Suppose the sufficiency holds for $k-1$ vectors $E=\{J_1(A),\dots,J_{k-1}(A)\}$. We show that it is true for k vectors $J_1(A),\dots,J_{k-1}(A),J_k(A)$. For this we need only to show that $\sum_{i=1}^{k} J_i(A,x)\leqslant CA(x)$ holds for all $x\in\{d-L_k(A)+1,d-L_k(A)+2,\dots,d\}$ because only at these points $J_k(A,x)\neq 0$ holds.

Note that $\sum_{i=1}^{k} J_i(A,x)\leqslant CA(x)$ holds for all $x\in\{d-PA(E)+2,\dots,d\}$ because the value $\sum_{i=1}^{k} J_i(A,x)$ increases at the most to one with time unit x. The considered inequality also holds for all $x\in\{d-L_k(A)+1,\dots,d-PA(E)\}$ because the sum $\sum_{i=1}^{k} J_i(A,x)$ either decreases by one if x decreases by one or it decreases by two if x decreases by two. Finally, $\sum_{i=1}^{k} J_i(A,x)\leqslant CA(x)$ holds for all $x\leqslant d-L_k(A)$ by the induction assumption. □

The same result can be stated and proved for the case $E=\{J_1(B),\dots,J_k(B)\}$ by the replacement of A by B in Theorem 2 and it's proof. The proved theorem shows that in order to check whether the sum of some vectors E exceeds vector C or not, we do not need to know all components of $\sum_{J_i\in E} J_i$, but only the values $\sum_{J_i\in E} J_i(A,d-PA(E)+1)$ and $\sum_{J_i\in E} J_i(B,d-PB(E)+1)$, where $PB(E)$ is defined analogous to $PA(E)$.

The following example illustrates the use of the theorem.

Example 3. Consider three jobs $E=\{J_1,\ J_2,\ J_3\}$ defined by the following data $n_1=7$, $d_1=7$, $\mu_{11}=A$, $n_2=5$, $d_2=5$, $\mu_{21}=B$ and $n_3=2$, $d_3=6$, $\mu_{31}=B$. The corresponding vectors are $J_1(A)=(1,1,2,2,3,3,4)$, $J_1(B)=(0,1,1,2,2,3,3)$, $J_2(A)=(0,1,1,2,2,2,2)$, $J_2(B)=(1,1,2,2,3,3,3)$, and $J_3(A)=(0,0,0,0,0,1,1)$, $J_3(B)=(0,0,0,0,1,1,1)$. Now we find that $PA(E)=2$ and $PB(E)=3$, so $\sum_{i=1}^{3} J_i(A,d-PA(E)+1)=\sum_{i=1}^{3} J_i(A,6)=6$ and $\sum_{i=1}^{3} J_i(B,d-PB(E)+1)=\sum_{i=1}^{3} J_i(B,5)=6$ hold. Therefore by Theorem 2 set E is not a feasible one because $\sum_{i=1}^{3} J_i(B,d-PB(E)+1)>CB(d-PB(E)+1)$ holds. Now shift all jobs one time unit to the right, i.e. set $d_1=8$, $d_2=6$ and $d_3=7$. In this case we find that $\sum_{i=1}^{3} J_i(A,7)=6$ and $\sum_{i=1}^{3} J_i(B,6)=6$. Therefore, now the set E is a feasible one.

In the next algorithm each set $\varphi_i(x_1,x_2,x_3,x_4,x_5)=\{(y_1,y_2),\ (y_1',y_2'),\dots\}$ represents a list of feasible subsets of $\{J_1,\dots,J_i\}$, where each pair (y_1,y_2) corresponds to some feasible subset E with the following meanings: $FA(E)=x_1$, $PA(E)=x_2$, $FB(E)=x_3$, $PB(E)=x_4$, $\sum_{J_j\in E} w_j=x_5$ and also $\sum_{J_j\in E} H_j(A)=y_1$ and $\sum_{J_j\in E} H_j(B)=y_2$. It is not hard to see that if for two possible elements (y_1,y_2) and (y_1',y_2') of $\varphi_i(x_1,x_2,x_3,x_4,x_5)$ the pair (y_1',y_2') exceeds pair (y_1,y_2), i.e. $y_1\leqslant y_1'$ and $y_2\leqslant y_2'$ hold, then the couple

(y'_1, y'_2) is redundant and it can be rejected from further consideration. This implies that we need to keep in mind only incomparable values of $\varphi_i(x_1,\dots,x_5)$, i.e. for any two values (y_1, y_2) and (y'_1, y'_2) inequalities $y_1 \leqslant y'_1$ and $y_2 \geqslant y'_2$ or $y_1 \geqslant y'_1$ and $y_2 \leqslant y'_2$ must hold. At Step 0 of the algorithm we suppose that there are no feasible subsets and set $\varphi_i(x_1,\dots,x_5)=\emptyset$ for all $i \in \{1,\dots,n\}$ and for all $(x_1,x_2,x_3,x_4,x_5) \in X$. Here X can be defined in the following way: $x_1 \in \{D_1(A),\dots,D_n(A)\}$, $x_2 \in \{L_1(A),\dots,L_n(A)\}$, $x_3 \in \{D_1(B),\dots,D_n(B)\}$, $x_4 \in \{L_1(B),\dots,L_n(B)\}$, $x_5 \in [0,\sum_{i=1}^{n} w_i]$, and $|x_1 - x_3| \in \{0,1\}$, $|x_2 - x_4| \in \{0,1\}$ hold. The last relations are caused by the fact that $|L_i(A) - L_i(B)| = 1$ and $|D_i(A) - D_i(B)| = 1$ hold for all $i \in \{1,\dots,n\}$. Therefore $|X| = O(n^2 \sum w_i)$ holds.

Algorithm.

Step 0: For each $i \in \{0,1,\dots,n\}$ do
For each $(x_1,x_2,x_3,x_4,x_5) \in X$ do
$\varphi_i(x_1,x_2,x_3,x_4,x_5):=\emptyset$

Step 1: For $i:=1$ to n do
Begin

Step 2: $\varphi_i(D_i(A),L_i(A),D_i(B),L_i(B),w_i):=\{(H_i(A),H_i(B))\}$

Step 3: For each $(x_1,x_2,x_3,x_4,x_5) \in X$ do
$\varphi_i(x_1,x_2,x_3,x_4,x_5):=\varphi_{i-1}(x_1,x_2,x_3,x_4,x_5) \cup \varphi_i(x_1,x_2,x_3,x_4,x_5)$

Step 4: For each $(x_1,x_2,x_3,x_4,x_5) \in X$ do
For each $(y_1, y_2) \in \varphi_{i-1}(x_1,x_2,x_3,x_4,x_5)$ do
Begin

Step 4.1: Arrange values $x_1,x_2,D_i(A),L_i(A)$ in nondecreasing order and denote by x'_1 and x'_2 the first and the second values correspondingly

Step 4.2: Arrange values $x_3,x_4,D_i(B),L_i(B)$ in nondecreasing order and denote by x'_3 and x'_4 the first and the second values correspondingly

Step 4.3: $x'_5:=x_5+w_i$

Step 4.4: Check whether $(y_1 + H_i(A) - \lceil (x'_2 - x'_1)/2 \rceil,\ y_2 + H_i(B) - \lceil (x'_4 - x'_3)/2 \rceil) \leqslant (CA(d - x'_2 + 1),\ CB(d - x'_4 + 1))$ holds. If it does then set $\varphi_i(x'_1,x'_2,x'_3,x'_4,x'_5):=\varphi_i(x'_1,x'_2,x'_3,x'_4,x'_5) \cup \{(y_1 + H_i(A),\ y_2 + H_i(B))\}$

Step 4.5: Among all values $\varphi_i(x'_1,x'_2,x'_3,x'_4,x'_5)$ leave only incomparable pairs
End
End.

The sets $\varphi_n(x_1,x_2,x_3,x_4,x_5) \neq \emptyset$ with the maximal value of x_5 calculated by the algorithm correspond to the optimal sets of early jobs.

In words the algorithm may be described as follows. At each iteration i of Step 1 the algorithm generates the set of feasible subsets of $\{J_1,\dots,J_i\}$, written by φ_i. At Step 2 it generates one element sets $\{J_i\}$. As it was supposed earlier any one element set is a feasible one. At Step 3 the algorithm includes all sets, written by φ_{i-1} and generated earlier, into φ_i. At each iteration of Step 4 for each set E described by φ_{i-1} the algorithm forms new set $E \cup \{J_i\}$ described by $\varphi_i(x'_1,x'_2,x'_3,x'_4,x'_5) \ni (y_1 + H_i(A), y_2 + H_i(B))$. At Steps 4.1–4.3 five values $x'_1 = FA(E \cup \{J_i\})$, $x'_2 = PA(E \cup \{J_i\})$, $x'_3 = FB(E \cup \{J_i\})$, $x'_4 = PB(E \cup \{J_i\})$ and $x'_5 = \sum_{J_j \in E \cup \{J_i\}} w_j$ are calculated. Now

we show that at Step 4.4 the algorithm checks feasibility of $E \cup \{J_i\}$. Consider the vector $\sum_{J_j \in E \cup \{J_i\}} J_j(A)$. By definition

$$\sum_{J_j \in E \cup \{J_i\}} J_j(A, d) = \cdots = \sum_{J_j \in E \cup \{J_i\}} J_j(A, d - FA(E \cup \{J_i\}) + 1) = y_1 + H_i(A)$$

holds. Then due to the structure of vector J_i we have the following:

$$\sum_{J_j \in E \cup \{J_i\}} J_j(A, d - FA(E \cup \{J_i\})) = \sum_{J_j \in E \cup \{J_i\}} J_j(A, d - FA(E \cup \{J_i\}) - 1) = y_1 + H_i(A) - 1$$

$$\sum_{J_j \in E \cup \{J_i\}} J_j(A, d - FA(E \cup \{J_i\}) - 2) = \sum_{J_j \in E \cup \{J_i\}} J_j(A, d - FA(E \cup \{J_i\}) - 3) = y_1 + H_i(A) - 2$$

$\cdots$

$$\sum_{J_j \in E\{J_i\}} J_j(A, d - PA(E \cup \{J_i\}) + 1) = y_1 + H_i(A) - \left\lceil \frac{x_2' - x_1'}{2} \right\rceil.$$

Therefore, by Theorem 2 the set $E \cup \{J_i\}$ is a feasible one if and only if $y_1 + H_i(A) - \lceil (x_2' - x_1')/2 \rceil \leqslant CA(d - PA(E \cup \{J_i\}) + 1)$ and $y_2 + H_i(B) - \lceil (x_4' - x_3')/2 \rceil \leqslant CB(d - PB(E \cup \{J_i\}) + 1)$ hold. Finally, at Step 4.5 we delete all redundant sets.

Next we estimate the complexity of the above algorithm. At Step 0 the equality $|X| = \mathrm{O}(n^2 \sum w_i)$ holds. Hence at this step we need $\mathrm{O}(n^3 \sum w_i)$ time. Step 1 looks over all values of i. There are only n possible values. Step 2 forms one set $\{i\}$. Step 3 looks over all elements of X. It takes $|X| = \mathrm{O}(n^2 \sum w_i)$ time. Step 4 looks over all elements (y_1, y_2) of $\varphi_{i-1}(x_1, x_2, x_3, x_4, x_5)$. Note that $|y_1 - y_2| \in \{0, 1, \ldots, n\}$. This is caused by the fact that $|H_i(A) - H_i(B)| \in \{0, 1\}$ holds and hence $|y_1 - y_2| = |\sum_{J_i \in E} H_i(A) - \sum_{J_i \in E} H_i(B)| \in \{0, 1, \ldots, |E|\}$ holds. For fixed $i, x_1, \ldots, x_5$ there are not more than $\mathrm{O}(n)$ incomparable pairs (y_1, y_2). Hence for fixed i not more than $\mathrm{O}(n^3 \sum w_i)$ various values (y_1, y_2) with various $(x_1, x_2, x_3, x_4, x_5)$ must be considered. Therefore Step 4 looks over $\mathrm{O}(n^3 \sum w_i)$ values. At Step 4.4 we compare the newly created pair $(y_1 + H_i(A), y_2 + H_i(B))$ with all pairs in $\varphi_i(x_1', x_2', x_3', x_4', x_5')$. It takes $\mathrm{O}(n)$ time. Hence for Step 4 we need $\mathrm{O}(n^4 \sum w_i)$ time.

Therefore the overall complexity of the algorithm is $\mathrm{O}(n^5 \sum w_i)$.

Now we prove NP-hardness for the problem $J2 \,|\, t_{ij} = 1 \,|\, \sum w_i U_i$. We do it in a manner similar to that for $1||\sum w_i U_i$ problem, see [5]. The proof is based on the reduction of $J2 \,|\, t_{ij} = 1 \,|\, \sum w_i U_i$ from the following Subset Sum problem: Given even positive integers $a_1, \ldots, a_n, b$, does there exist a subset $S \subset T = \{1, \ldots, n\}$ such that $\sum_{j \in S} a_j = b$?

Given positive integers $a_1, \ldots, a_n, b$, the following instance of $J2 \,|\, t_{ij} = 1 \,|\, \sum w_i U_i$ can be constructed. There are n jobs J_i, $i \in \{1, \ldots, n\}$, with $\mu_{i1} = A$, $n_i = 2a_i$, $d_i = 2b$, $w_i = 2a_i$ and one more additional job J_0 with $\mu_{01} = B$, $n_0 = 2\sum_{i=1}^{n} a_i$, $w_0 = 2b$, $d_0 = 2\sum_{i=1}^{n} a_i$.

A schedule with $\sum w_i U_i \geqslant 4b$ exists if and only if there exists a subset $T \in \{1,\ldots,n\}$ such that $\sum_{i\in T} a_i = b$.

So problem $J2 \mid t_{ij} = 1 \mid \sum w_i U_i$ is NP-hard in the ordinary sense.

4. Conclusion

The algorithm in the previous section finds a maximum-weighted feasible set of jobs in $O(n^5 \sum w_i)$ time. To schedule a feasible set in a feasible way we can use Brucker's algorithm [1] which takes $O(\sum n_i)$ time. However Timkovsky [12] schedules a feasible set in $O(n^2)$ time. Therefore with the help of the algorithm problem $J2 \mid t_{ij}=1 \mid \sum w_i U_i$ can be solved in $O(n^5 \sum w_i)$ time and problem $J2 \mid t_{ij}=1 \mid \sum U_i$ can be solved in $O(n^6)$ time.

A good survey of unit-time job shop scheduling problems can be found in [13]. We point out only that the problem $J2 \mid t_{ij}=1 \mid \sum C_i$ can be solved optimally in $O(n \log n)$ time, see [8,6].

Acknowledgements

We would like to thank the referees for their constructive comments. The idea of the presented proof of Theorem 1 comes from a referee.

References

[1] P. Brucker, A linear time algorithm to minimize maximum lateness for the two-machine unit-time, job-shop, scheduling problem, Lecture Notes in Control and Inform. Sci. 38 (1982) 566–571.
[2] P. Brucker, Minimizing maximum lateness in a two-machine unit-time job-shop, Computing 27 (1981) 367–370.
[3] T. Gonzalez, S. Sahni, Flow shop and job shop schedules: complexity and approximation, Oper. Res. 26 (1978) 36–52.
[4] N. Hefetz, I. Adiri, An efficient optimal algorithm for the two-machine, unit-time, job-shop, schedule-length problem, Math. Oper. Res. 7 (1982) 354–360.
[5] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J.W. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York, 1972, pp. 85–103.
[6] S.A. Kravchenko, Optimal scheduling for two-machine unit-time job-shop problems, Preprint N 7, Institute of Engineering Cybernetics, Minsk, Belarus, 1995 (in Russian).
[7] W. Kubiak, S. Sethi, C. Sriskandarajah, An efficient algorithm for a job shop problem, Math. Indust. Systems 1 (1995) 203–216.
[8] W. Kubiak, V.G. Timkovsky, Total completion time minimization in two-machine job shop with unit-time operations, European J. Oper Res. 94 (1996) 310–320.
[9] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, Sequencing and scheduling:algorithms and complexity, in: S.C. Graves, A.H.G. Rinnooy Kan, P. Zipkin (Eds.), Handbooks in Operations Research and Management Science, Vol. 4: Logistic of Production and Inventory, North-Holland, Amsterdam, 1993, pp. 445–522.

[10] J.K. Lenstra, A.H.G. Rinnooy Kan, Computational complexity of discrete optimization problems, Ann. Discrete Math. 4 (1979) 121–140.
[11] V.G. Timkovsky, Polynomial-time algorithm for Lenstra-Rinnooy Kan two-machine scheduling problem, Kibernetika 2 (1985) 109–111 (in Russian).
[12] V.G. Timkovsky, A polynomial-time algorithm for the two-machine unit-time release-date job-shop schedule-length problem, Discrete Appl. Math. 77 (1997) 185–200.
[13] V.G. Timkovsky, Is a unit-time job shop not easier than identical parallel machines, Discrete Appl. Math. 85 (1998) 149–162.